WIP: End-to-End Analysis of Adversarial Attacks to Automated Lane Centering Systems

Hengyi Liang*, Ruochen Jiao*, Takami Sato†, Junjie Shen†, Qi Alfred Chen†, Qi Zhu
Northwestern University † University of California, Irvine

***Abstract*—Machine learning techniques, particularly those based on deep neural networks (DNNs), are widely adopted in the development of advanced driver-assistance systems (ADAS) and autonomous vehicles. While providing significant improvement over traditional methods in average performance, the usage of DNNs also presents great challenges to system safety, especially given the uncertainty of the surrounding environment, the disturbance to system operations, and the current lack of methodologies for predicting DNN behavior. In particular, adversarial attacks to the sensing input may cause errors in systems' perception of the environment and lead to system failure. However, existing works mainly focus on analyzing the impact of such attacks on the sensing and perception results and designing mitigation strategies accordingly. We argue that as system safety is ultimately determined by the actions it takes, it is essential to take an end-to-end approach and address adversarial attacks with the consideration of the entire ADAS or autonomous driving pipeline, from sensing and perception to planing, navigation and control. In this paper, we present our initial findings in quantitatively analyzing the impact of a type of adversarial attack (that leverages road patch) on system planning and control, and discuss some of the possible directions to systematically address such attack with an end-to-end view.**

I. Introduction

Recent years witnessed significant advancement of autonomous driving techniques and advanced driver-assistance systems (ADAS), with widely-adopted applications such as Adaptive Cruise Control (ACC), Forward Collision-Avoidance Assistance (FCA), and Automated Lane Centering (ALC). Such systems typically consist of a pipeline of sensing, perception, planning, and control modules. They rely on on-board sensors to collect data at real time, use perception algorithms to process the data and understand the environment, and then make planning and control decisions for the vehicle. In these systems, deep neural networks (DNNs) are used prevalently, especially for perception tasks such as lane detection and object detection, due to their high level of average accuracy. Moreover, there have also been increasing application of DNNs to other modules, such as prediction, planning, and control.

However, the safety and security of these learning-based ADAS and autonomous systems can hardly be guaranteed or even quantitatively analyzed, given the various uncertainties in the system and its surrounding environment, such as the inherent uncertainties from the dynamic environment, the disturbances to system operations from environment interference, transient errors, malicious attacks, and the current lack of methodologies for predicting the DNN behavior [1], [2].

In the literature, there has been a large body of works demonstrating the vulnerability of DNNs to external disturbances and adversarial attacks [3]–[6]. In particular, some recent works try to attack DNN-based ADAS and autonomous systems by adding adversarial perturbations to the physical environment in a stealthy way [7], [8]. The adversarial patterns are carefully designed so that they can make the perception module render wrong results while themselves can hardly be noticed by human – one such example is the dirty road patch attack shown in [8]. The wrong perception results caused by these attacks will then affect the downstream planning, navigation, and control decisions, and possibly cause dangerous situations and accidents (e.g., causing victim vehicle to drive off the road or collide with other objects).

Most prior defense works either focus on 1) making the neural networks themselves more robust against adversarial attacks [9], [10] or 2) trying to detect anomaly in the input data [11], [12]. While it is clear that the impact of those adversarial attacks on system safety has to be eventually reflected through vehicle planning, navigation, and control, there is currently a lack of *systematic and quantitative* methods to analyze such impact throughout the end-to-end ADAS and autonomous driving pipeline and to design mitigation strategies accordingly. In this work, we focus on the safety and robustness of the entire system and explore holistic end-to-end methods for the analysis and mitigation of adversarial attacks to driving assistance systems. We analyze how the perturbation from sensor data propagates through the pipeline and affect the final control actions (e.g., steering decisions). Our overall goals in this area include the following:

1) Analyzing the impact of each module on system safety and identifying proper interfaces between modules that can be utilized to improve system safety and robustness.
2) Proposing a system-level framework to improve the safety of DNN-based driving assistance systems, e.g., by quantifying the perception uncertainty and adapting planning and control accordingly.
3) Evaluating our proposed approach on a production-grade simulation environment (e.g., OpenPilot [13] combined with LGSVL [14]).

This work-in-progress paper mainly focuses on the first goal. We analyze the ALC system as a representative ADAS application, identify how the uncertainty/confidence changes under various adversarial scenarios from the dirty road patch attack, and discuss potential methods to improve system safety and robustness.

II. System Model

In this paper, we use the ALC system as a demonstrating example. We believe that our analysis methodology could

* These two authors contribute equally to the work.

Workshop on Automotive and Autonomous Vehicle Security (AutoSec) 2021
25 February 2021, Virtual
ISBN 1-891562-68-1
https://dx.doi.org/10.14722/autosec.2021.23034
www.ndss-symposium.org

also be applied to other autonomous driving functionalities (e.g., localization), where there are uncertainties from external environment. The ALC is a popular Level 2 autonomous driving system that is widely available for modern vehicles. For example, many OEM vehicles such as Mercedes-Benz C-class, Tesla and GM CT6 come with lane centering feature. Moreover, there are also third party open-source autonomous driving software stacks (e.g., OpenPilot, Baidu Apollo, Autoware) that integrate ALC.

Figure 1 shows an ALC system structure that involves three major modules: *perception*, *planning* and *control*. The perception module takes sensor data as input and then detects lane lines and predicts any other objects on the road (e.g., vehicles, obstacles). The planning module mainly calculates the desired trajectory based on the output of perception module. The control module then calculates a desired steering angle and speed that will be applied to the vehicle actuators. In the following, we use the ALC system in OpenPilot as an example and briefly introduce each module.

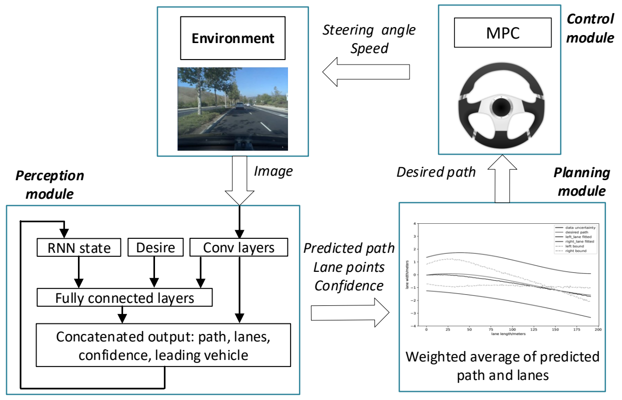

Fig. 1: The end-to-end pipeline of automated lane centering.

A. DNN-based Lane Detection Module

Traditional perception module uses edge detection algorithms to detect lane lines from images captured by the front camera [15]. In recent years, DNN-based lane detection models have achieved the state-of-the-art accuracy and been widely adopted in production-level ALC systems. Since lane lines do not change significantly across consecutive frames, recurrent neural networks (RNNs) ares typically used to make the prediction more stable. The perception module in Figure 1 highlights the input-output relation of the OpenPilot lane detection model. Particularly, a convolutional neural network (CNN) is used to process current image data and then a combination of the CNN output and the last RNN state is fed to a fully connected network. OpenPilot lane detection model leverages Multiple Hypotheses Prediction (MHP) [16] to capture ambiguity in the prediction. The output of the lane detection model contains 1) 192 predicted lane line points, in the form of mean and standard variance, for each lane, 2) the confidence value for each predicted lane line, 3) a predicted driving path, and 4) information related to the leading vehicle. After the lane detection inference, a post-process is followed to fit the points into lane line curves. Specifically, in OpenPilot, the weighted least square regression method is used.

B. Planning Module

The planning module generates the desired path for the next period. For ALC, this means that the desired path should be the center of the detected left lane and right lane and the steering angle should keep the vehicle following the desired path. As mentioned earlier, the perception module handles the ambiguity of prediction by outputting confidence scores for the left lane and the right lane. In the OpenPilot planning module, this information is used to generate the final desired path as a weighted sum of the predicted lanes and the predicted path. Intuitively, if the prediction is less confident on the predicted lanes, then the final desired path relies on the predicted path; otherwise, if the prediction has high confidence on the predicted left lane and right lane, the final desired path will be close to the center of the predicted left lane and right lane.

C. Control Module

The main goal of the control module is to calculate the vehicle maneuver for the next period and convert it into a command that will be applied to vehicle actuators, e.g., steering, throttle, and brake command to the chassis. For lateral control, after obtaining the desired path from the path planner, a controller is used to calculate the corresponding steering angle based on the desired path. In OpenPilot, model predictive control (MPC) is used to calculate the steering angle. The constraints include 1) simplified system dynamics, 2) vehicle heading constraints, and 3) maximum steering angle constraints. In our simulation environment, due to the constraint of the mechanical unit, we consider that the steering rate is limited by a maximum value. The throttle is generated by a PID controller by setting an appropriate reference speed.

III. Threat Model

In this paper, we mainly consider attacks achieved through physical domain. Specifically, we assume that attacker cannot hack into the victim vehicle and should not be able to modify the software of the victim vehicle directly. However, the attacker has the ability to modify the physical environment that can be captured by the sensors equipped on the victim vehicle (e.g., cameras). The attacker's goal is to carefully change the appearance of certain object such that 1) the object can hardly be noticed by human driver, 2) and it can cause the ADAS or autonomous driving system to drive off the road. There are many adversarial attacks against deep learning based tasks such as image classification and object detection, but we focus on attacks specially targeting deep learning based driving assistance systems so as to analyze and evaluate the safety of those systems. For example, the work in [7] can generate an adversarial billboard to cause steering angle error; the work in [8] is able to generate a gray scale dirty patch on road such that vehicles passing through the patch will deviate (either left or right). In the following experiments, we use the dirty road patch attack as a case study, and we believe that our approach may be extended for other types of physical world attacks.

In fact, the optimization-based physical attack in [8] is the first attack that is systematically designed for the ALC system. Specifically, the attackers first collect the input image and predicted steering angle of the ALC on a certain road. Using these images, the knowledge of the ALC system, and the kinematic model of the vehicle, they can optimize an adversarial patch to make the vehicle deviate from the original trajectory to the greatest extent. In the experiment, it can lead the vehicle out of the lane boundary within 1 second (which means the lateral deviation is larger than 0.735m in highway) with high success rate.

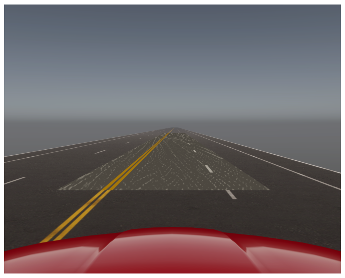

Fig. 2: Image of the adversarial patch captured by front camera.

IV. Experiments and Observations

As a preliminary case study, we perform the ALC attack in a simulation environment. We bridge the product-level simulator LGSVL with OpenPilot and put a carefully designed patch on the road [8]. During each simulation step, we obtain the image frame captured by the front camera (a demo frame is shown in Figure 2) and the vehicle status through the API provided by the LGSVL simulator. We then pass these data to OpenPilot. OpenPilot ALC will calculate a desired steering angle for the next simulation step. Then, the steering command is sent back to LGSVL through the API.

To evaluate the effectiveness of the attack, we conduct a series of experiments with different settings of the patch. Figure 3 shows the trajectory of the ego vehicle when we put the patch on different locations of the road. The maximum lateral deviation is over 1 meter, which will cause the vehicle to drive into the opposite lane. Although the ego vehicle finally drives back to its original lane after it has passed through the dirty patch area, this behavior is extremely dangerous. Then, we gradually reduce the perturbation area of the patch. Here, the perturbation area ratio (PAR) denotes the percentage of the pixels on the patch that can be perturbed. For example, 30% PAR means that only 30% of pixels can be perturbed while the rest 70% pixels have the same color as the road. Figure 4 shows the trajectory of the ego vehicle under different level of PAR. The ego vehicle drives off the road when PAR is 50% or 30%. When PAR is 10%, the ego vehicle can stay in the lane but the patch still can impact its stability.

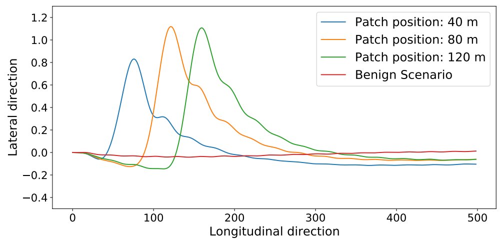

Fig. 3: The trajectory of the vehicle when the dirty patch is placed on different locations of the road.

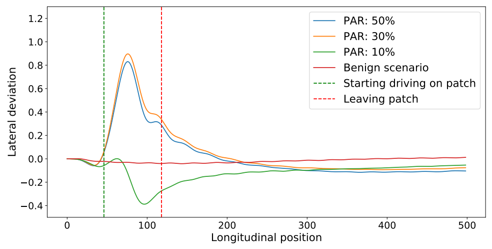

Fig. 4: The trajectory of the vehicle when the dirty patch is designed with different values of the PAR.

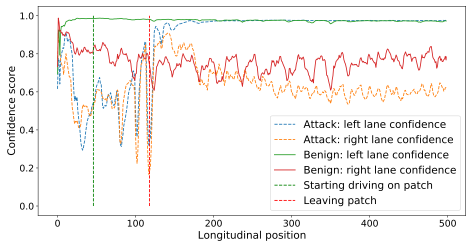

Fig. 5: The confidence of the perception module drops significantly when the ego vehicle is approaching the patch.

To study how the adversarial attack can lead the vehicle to deviate, we inspect the interface between each module and try to reason about how the impact of the adversarial attack can propagate through each module.

Perception Confidence: We first check the result of the perception module. In OpenPilot, the MHP model is trained with binary cross entropy loss, which means the output can indicate the confidence in whether there are lane lines [17]. Through our experiments with different settings of the patch, we notice a general phenomenon: the confidence scores for both left lane and right lane drop significantly when the vehicle is approaching the patch, as shown in Figure 5, while the confidence score will almost stay stable under benign situation or ineffective attack. The dashed green line in Figure 5 indicates the starting point of the patch. As we can see, the confidence score drops even before the vehicle is on the patch (e.g., when the vehicle is at around 30 meters). However, when the confidence score drops, the vehicle has not started to deviate. This observation indicates that the impact of adversarial attack comes earlier than when the driver can perceive.

Biased Desired Path: We then inspect the result of path planner and try to figure out how the perception confidence may affect the final desired trajectory. Figure 6 is the result of one frame when the vehicle is approaching the patch. The predicted left lane, predicted right lane and the predicted path are the results generated by the perception module. As we can see, these three curves lean to the left (i.e., the positive direction of the y axis). If the vehicle just follows the predicted path, it will definitely drive into the opposite lane. By analyzing the code of the path planner, we find that OpenPilot tries to leverage the confidence scores as weights. The final desired path can be considered as a weighted sum of the predicted left lane, predicted right lane, and the predicted path. However, as we observe that the confidence score for predicted lanes drops significantly, the final desired path will largely depend on the predicted path, which is bent to wrong

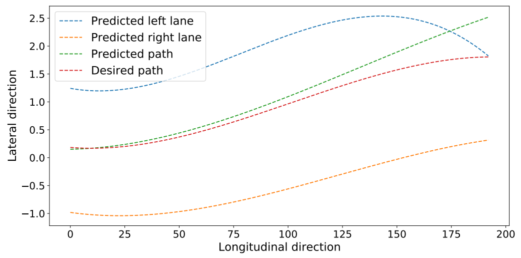

Fig. 6: Desired trajectory generated by OpenPilot path planner deviates towards left when the vehicle is approaching the patch.

direction. Since the desired path is biased, the MPC controller cannot calculate a correct steering angle for the next step.

Potential Mitigation Methods: Based on the preliminary experiments and observations, we find that the confidence score can work as a triggering signal to indicate the presence of adversarial attack or anomalous noises in this case. We believe that an end-to-end approach is needed to mitigate the attacks, and there are some potential directions:

1) We may try to quantify/estimate the uncertainty of the DNN-based perception module, including both epistemic uncertainty (caused by unbalanced distribution of the training data) and aleatory uncertainty (caused by noises of input data). Such estimated uncertainty can then be used by the planning and control module to improve system safety.
2) We may develop an adaptation strategy to holistically adjust each module to mitigate the impact of adversarial attacks.
3) In the perception module, we may use other sensors' data (e.g., from LiDAR and GPS) to compensate the result of cameras, especially when the confidence is low.
4) We may leverage the temporal and spatial correlation of consecutive inputs to detect the anomaly of current sensor data or even re-use more reliable results from recent data samples. Besides, the RNN can be modified to make better use of the temporal correlation and assign more weights on frames with high confidence and less uncertainty.

V. Conclusion and Future Work Plan

We argue that it is essential to address adversarial attacks on driving assistance systems from an end-to-end perspective, by quantitatively analyzing the impact of those attacks across the ADAS or autonomous driving pipeline and designing mitigation strategies in a holistic manner. In our preliminary work, we studied an ALC system, analyzed how different modules interact under dirty road patch attacks, and identified a sensitive signal that can be leveraged to detect those attacks. We are currently working on better quantifying the uncertainty of the perception module and using it to guide the adaption of the planning and control module for improving system safety. In future, we also plan to study other ADAS and autonomous driving applications and try to generalize our approach.

Acknowledgment

We gratefully acknowledge the support from CNS-1839511, CNS-1834701, IIS-1724341, CNS-1850533, CNS-1929771, CNS-1932464, USDOT UTC Grant 69A3552047138, and ONR grant N00014-19-1-2496.

References

[1] Q. Zhu, W. Li, H. Kim, Y. Xiang, K. Wardega, Z. Wang, Y. Wang, H. Liang, C. Huang, J. Fan, and H. Choi, "Know the unknowns: Addressing disturbances and uncertainties in autonomous systems : Invited paper," in *ICCAD*, 2020, pp. 1–9.

[2] Q. Zhu, C. Huang, R. Jiao, S. Lan, H. Liang, X. Liu, Y. Wang, Z. Wang, and S. Xu, "Safety-assured design and adaptation of learning-enabled autonomous systems," *ASP-DAC*, 2021.

[3] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," 2017.

[4] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, 2016, pp. 372–387.

[6] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] H. Zhou, W. Li, Z. Kong, J. Guo, Y. Zhang, B. Yu, L. Zhang, and C. Liu, "Deepbillboard: Systematic physical-world testing of autonomous driving systems," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ser. ICSE '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 347–358.

[8] T. Sato, J. Shen, N. Wang, Y. J. Jia, X. Lin, and Q. A. Chen, "Hold tight and never let go: Security of deep learning based automated lane centering under physical-world attack," *arXiv preprint arXiv:2009.06701*, 2020.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[10] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 582–597.

[11] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018, pp. 7167–7177.

[12] Z. Zheng and P. Hong, "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018, pp. 7913–7922.

[13] "Openpilot," https://comma.ai/.

[14] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, E. Agafonov, T. H. Kim, E. Sterner, K. Ushiroda, M. Reyes, D. Zelenkovsky, and S. Kim, "Lgsvl simulator: A high fidelity simulator for autonomous driving," 2020.

[15] J. B. McDonald, "Application of the hough transform to lane detection and following on high speed roads," in *in Motorway Driving Scenarios", in Proceeding of Irish Signals and Systems Conference*, 2001.

[16] C. Rupprecht, I. Laina, R. DiPietro, M. Baust, F. Tombari, N. Navab, and G. D. Hager, "Learning in an uncertain world: Representing ambiguity through multiple hypotheses," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3591–3600.

[17] D. Ramos, J. Franco-Pedroso, A. Lozano-Diez, and J. Gonzalez-Rodriguez, "Deconstructing cross-entropy for probabilistic binary classifiers," *Entropy*, vol. 20, no. 3, p. 208, 2018.